DELFT UNIVERSITY OF TECHNOLOGY

REPORT 08-13

Numerical Methods for Industrial Flow Problems

Ibrahim, C. Vuik, F. J. Vermolen, and D. Hegen

ISSN 1389-6520

Reports of the Department of Applied Mathematical Analysis

Delft 2008

NUMERICAL METHODS FOR INDUSTRIAL FLOW PROBLEMS

IBRAHIM[1], C. VUIK, F. J. VERMOLEN, AND D. HEGEN

ABSTRACT. In this report, Partial Differential Equations (PDEs) are solved numerically. These include Poisson Equation, Convection-Diffusion Equation, either steady state or time-varying, and Burgers' Equation. Methods used are Standard Galerkin Algorithm (SGA) and Streamline Upwind Petrov-Galerkin. For the nonlinear Partial Differential Equations, Picard Iteration is combined with Finite Element Methods. For time-varying PDEs, Explicit, Implicit and IMEX schemes are used. A system of nonlinear equations is solved by applying the above mentioned methods.

1. INTRODUCTION

Mathematical modeling of many physical systems is often described by so called conservation equations. In general, such equations are represendted by nonlinear (coupled) partial differential equations. Analytical solutions are hard or impossible to find for many PDEs. Experimental data are not available all the time. Numerical Methods can be used even in such cases and sometimes it is the only choice. Error estimation, solution convergence and stability are important issues in numerical methods.

2. FINITE ELEMENT METHOD

The finite element method is a numerical technique, used to find approximate solutions of partial differential equations. It is more general, as compared to the finite difference and the finite volume methods in the sense that, it can handle complex geometries and it is not restricted to differential equations in conservative form. The main features of the finite element are as follows. The domain of the problem is represented by a collection of simple subdomains, called elements. The collection of these elements is called the finite element mesh. Over each element, a property u described by a partial differential equation is approximated and represented by a sum of polynomials with coefficients u_j, which are found from a set of algebraic equations.

2.1. **Standard Galerkin Approximation.**

We will use an example to explain the standard Galerkin method. Consider the Poisson equation in one dimension:

$$-D\frac{d^2u}{dx^2} = f(x),\ 0 \leq x \leq 1, \tag{1}$$

$$u(0) = 2,\ u(1) = 1. \tag{2}$$

Multiply equation (1) by a test function $v(x)$ and integrate it over the computational domain, $[0,\ 1]$.

$$-D\int_0^1 \frac{d^2u(x)}{dx^2}v(x)dx = \int_0^1 f(x)v(x)dx. \tag{3}$$

To get the weak formulation, apply integration by parts.

$$-D\left[\frac{du}{dx}v(x)\right]_0^1 + D\int_0^1 \frac{du}{dx}\frac{dv}{dx}dx = \int_0^1 f(x)v(x)dx. \tag{4}$$

[1]The author is indebted to HEC, Pakistan and NUFFIC, The Netherlands for their financial and other support.

We choose $v(x)$ such that integrals in equation (4) exist. Furthermore, we take $v(0) = 0$ and $v(1) = 0$. Therefore, the boundary term in the above equation vanishes. Now we discretize $x,\ u(x)$ and $v(x)$ in the following way. The computational domain $x \in [0, 1]$ is divided into n finite intervals by considering $n + 1$ nodes, $x_0 = 0,\ x_1,\ x_3,\ ...,\ x_n = 1$. Each such interval is called an element. All elements are taken with equal length $h = \frac{1}{n}$. The *ith* element e_i is the interval from x_{i-1} to x_i. The unknown variable $u(x)$ is approximated by $\underline{u} = \{u_j\}$, where $u_j \approx u(x_j)$, and $j = 0,\ 1,\ ...,\ n$. Now $u(x)$ is written in terms of basis functions, $\phi_j(x)$, as follows,

$$u(x) \approx \sum_{j=0}^{n} u_j \phi_j(x), \tag{5}$$

where $u_0 = u(0)$ and $u_n = u(1)$ are known. We choose the function $v(x)$ as,

$$v(x) = \phi_i(x),\ i = 1, 2, ..., n-1. \tag{6}$$

Now $\phi_i(x)$ is a piecewise linear function defined as:

$$\phi_i(x) = \begin{cases} \frac{x-x_{i-1}}{x_i-x_{i-1}}, & \text{for } x \in [x_{i-1},\ x_i], \\ \frac{x-x_{i+1}}{x_i-x_{i+1}}, & \text{for } x \in [x_i,\ x_{i+1}], \\ 0, & \text{for } x \notin [x_{i-1},\ x_{i+1}]. \end{cases} \tag{7}$$

$$\phi_0(x) = \begin{cases} \frac{x-x_1}{x_0-x_1}, & \text{for } x \in [x_0,\ x_1], \\ 0, & \text{for } x \notin [x_0,\ x_1]. \end{cases}.$$

$$\phi_n(x) = \begin{cases} \frac{x-x_{n-1}}{x_n-x_{n-1}}, & \text{for } x \in [x_{n-1},\ x_n], \\ 0, & \text{for } x \notin [x_{n-1},\ x_n]. \end{cases}.$$

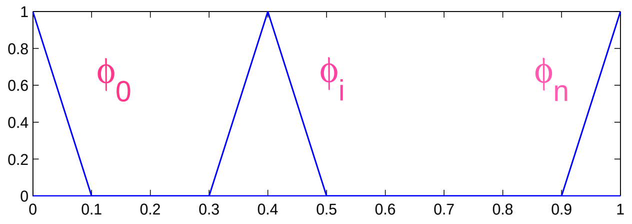

FIGURE 1. The piecewise linear basis function: $\phi_i(x)$

$$\phi_i(x_j) = \begin{cases} 0, & \text{for } j \neq i, \\ 1, & \text{for } j = i. \end{cases}$$

With this discretization, equation (1) takes the form

$$D \int_0^1 \sum_{j=0}^{n} u_j \frac{d\phi_j}{dx} \frac{d\phi_i}{dx} dx = \int_0^1 f(x)\phi_i(x) dx,\ i = 1, 2, ..., n-1, \tag{8}$$

or

$$D \sum_{j=1}^{n-1} u_j \int_0^1 \frac{d\phi_j}{dx} \frac{d\phi_i}{dx} dx = \int_0^1 f(x)\phi_i(x) dx + b_i,\ i = 1, 2, ..., n-1. \tag{9}$$

Where b_i is the ith element of vector $\underline{b}$ which incorporates boundary conditions and it is given as,

$$b_i = \begin{cases} -D\int_{x_0}^{x_1} u_0 \frac{d\phi_0}{dx}\frac{d\phi_1}{dx}dx, & \text{for } i = 1, \\ -D\int_{x_{n-1}}^{x_n} u_n \frac{d\phi_n}{dx}\frac{d\phi_{n-1}}{dx}dx, & \text{for } i = n-1, \\ 0, & 1 < i < n-1, \end{cases}$$

or,

$$b_i = \begin{cases} u_0\frac{D}{h}, & \text{for } i = 1, \\ u_n\frac{D}{h}, & \text{for } i = n-1, \\ 0, & 1 < i < n-1. \end{cases}$$

All the integrals on the left-hand side of equation (9) can be computed analytically. For $\int_0^1 f(x)\phi_i(x)dx$, we will use a numerical technique, if it is hard to determine otherwise. In order to implement equation (9) to a computer program, we use the following strategy. Equation (9) can be written in a matrix form:

$$S\underline{u} \;=\; \underline{f} \;+\; \underline{b}, \tag{10}$$

where S is an $n-1$ by $n-1$ matrix, called the stiffness matrix, $\underline{u}$ is the vector consisting of u_j, $j = 1, 2, ..., n-1$ and $\underline{f}$ is a vector of length $n-1$. The elements of S and $\underline{f}$ are given by,

$$S_{i,j} = D\int_0^1 \frac{d\phi_i}{dx}\frac{d\phi_j}{dx}dx,$$

$$f_i = \int_0^1 f(x)\phi_i(x)dx.$$

In order to find the matrix S, we calculate the element matrix S_e,

$$S_e = D\left[\begin{array}{cc} \int_{x_{i-1}}^{x_i} \frac{d\phi_{i-1}}{dx}\frac{d\phi_{i-1}}{dx}dx & \int_{x_{i-1}}^{x_i} \frac{d\phi_i}{dx}\frac{d\phi_{i-1}}{dx}dx \\ \int_{x_{i-1}}^{x_i} \frac{d\phi_{i-1}}{dx}\frac{d\phi_i}{dx}dx & \int_{x_{i-1}}^{x_i} \frac{d\phi_i}{dx}\frac{d\phi_i}{dx}dx \end{array}\right], \tag{11}$$

or,

$$S_e = \frac{D}{h}\left[\begin{array}{cc} 1 & -1 \\ -1 & 1 \end{array}\right]. \tag{12}$$

For the current problem, we have $D = 1,\ \Delta x = h = 0.1,\ n = 10,\ f(x) = 1,\ u_0 = 1,\ u_n = 0$. Initially S is an $n+1$ by $n+1$ matrix filled with zeros for all elements. With an index k, running from 1 to n, the stiffness matrix S is updated as:

$$S_{k-1,k-1} = S_{k-1,k-1} + S_{e_{1,1}},$$

$$S_{k-1,k} = S_{k-1,k} + S_{e_{1,2}},$$

$$S_{k,k-1} = S_{k,k-1} + S_{e_{2,1}},$$

$$S_{k,k} = S_{k,k} + S_{e_{2,2}}.$$

Before using S in further calculations, we omit its first and the last row as well as the first and the last column because we need to find only $n-1$ variables, u_i. For the integral on the right hand side, we can use a Newton-Cotes formula (the trapezoidal rule),

$$\int_{x_1}^{x_2} f(x)dx \approx \frac{x_2 - x_1}{2}(f(x_2) + f(x_1)). \tag{13}$$

Of course, the integral in our case is simple enough to calculate analytically. Now the element vector for $\underline{f}$ is given as:

$$\underline{f_e} = \left[\begin{array}{c} \int_{e_i} f(x)\phi_{i-1}dx \\ \int_{e_i} f(x)\phi_i dx \end{array}\right]. \tag{14}$$

Again an index k, running from 1 to n, helps to update the vector $\underline{f}$, initially loaded with zeros only.

$$f_{k-1} = f_{k-1} + f_{e_1},$$
$$f_k = f_k + f_{e_2}.$$

The first and the last elements of $\underline{f}$ are not considered. Hence we are left with $n-1$ elements. Now the approximate solution vector u_j can be determined, once we have $S, \underline{f}$ and $\underline{b}$,

$$\underline{u} = S^{-1}\underline{f} + S^{-1}\underline{b}. \tag{15}$$

Once we have the element matrix, in this case S_e, assembly of the corresponding matrix S is the same for all proceeding examples. Except that, if a boundary is not subject to an essential boundary condition, we will no longer remove the corresponding row and column of the stiffness matrix. A natural boundary condition applies in this case. The vector $\underline{b}$ and $\underline{f}$ are treated in a similar way. The equation (15) has been implemented in MATLAB. The result is shown in Figure 2, along with the relative difference with the exact solution, $u(x) = \frac{x^2}{2} - \frac{3}{2}x + 2$. We observe that for such a simple problem, there is (almost) no error i.e., $u(x) = u_j$ at node points $x = x_j$. In this example, boundary conditions were specified explicitly for

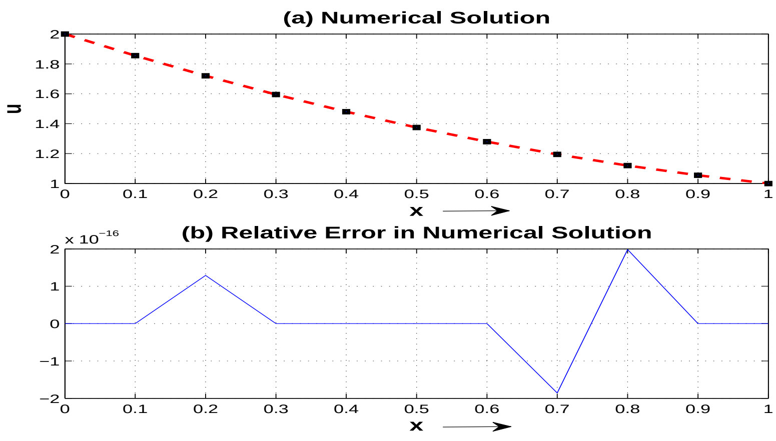

FIGURE 2. (a) The solution of Poisson equation by SGA, $D = 1,\ f = 1,\ u(0) = 2,\ u(1) = 1$, (b) Relative Error $\frac{u - u(x_j)}{u(x_j)}$, $u(x) = \frac{x^2}{2} - \frac{3}{2}x + 2$.

both ends. We can have other type of boundary conditions. Among many, few types are mentioned below.

Dirichlet Boundary Conditions: If boundary values are given explicitly, they are called Dirichlet Boundary conditions. e.g., $u(0) = 1,\ u(1) = a$ etc. For our purpose they are also called Essential Boundary Conditions.

Neumann Boundary Condition: It is the derivative of the unknown variable given at certain boundary point, e.g., $\frac{du}{dx} = 0$ at $x = 1$.

Robin Boundary Condition: It is a combination of above two boundary conditions. e.g., $\frac{du}{dx} + \sigma u = a,\ \sigma > 0,\ x = 1$.

For second order partial differential equations, the last two conditions are also called natural boundary conditions. If we do not specify boundary condition for one end, say at $x = 1$, then $\frac{du}{dx} = 0$ for $x = 1$ applies by default [2]. Of course it is true only approximately for a numerical solution. In this case we no longer omit the last column and row from S, $\underline{f}$ and $\underline{b}$. Furthermore $b_{n-1} = b_n = 0$. The result for this new condition is shown in Figure 3, with $u(0) = 1$.

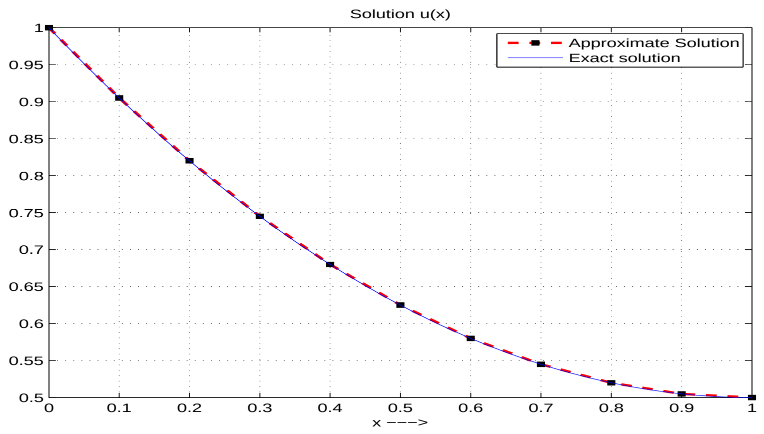

FIGURE 3. The solution of the diffusion equation by (red) SGA, $D = 1$, $f = 1$, $u(0) = 1$, $\frac{du(1)}{dx} = 0$, (blue) The exact solution, $u(x) = \frac{x^2}{2} - x + 1$.

3. Convection Diffusion Problem

In this section, we considered the steady state convection diffusion equation. Initially, we applied Standard Galerkin method to compute the approximate solution. When convection dominates over diffusion, we face undersireable wiggles in the numerical solution. We applied another method to overcome this difficulty.

$$
\begin{aligned}
-\epsilon \frac{d^2u}{dx^2} + c\frac{du}{dx} &= 0, \;\; 0 < x < 1, \qquad (16)\\
u(0) &= 1,\\
u(1) &= 0,
\end{aligned}
$$

where c is speed of the wave and ϵ is the diffusion coefficient. Both c and ϵ are assumed to be constant. Function u could be the temperature or some other physical quantity. In order to solve this equation, we apply the Standard Galerkin approach. Multiply equation(16) by a test function $v(x)$ and integrate over the problem domain.

$$
-\epsilon \int_0^1 \frac{d^2u}{dx^2} v(x)dx + c\int_0^1 \frac{du}{dx} v(x)dx = 0. \qquad (17)
$$

Integration by parts for the first integral on the left-hand side gives the following result.

$$
-\epsilon \left[\frac{du}{dx} v(x)\right]_0^1 + \epsilon \int_0^1 \frac{du}{dx}\frac{dv}{dx}dx + c\int_0^1 \frac{du}{dx} v(x)dx = 0.
$$

Again, choosing $v(0) = v(1) = 0$ makes the first term equal to zero. Discretization used for $u(x)$ is given by,

$$
u \approx \sum_{j=0}^{n} u_j \phi_j(x),
$$

where $u_0 = u(0)$ and $u_n = u(1)$, are known. The definition of the basis functions $\phi_j(x)$ and $v(x)$ is the same as given in the previous section. Equation (17) can be written as,

$$
\epsilon \int_0^1 \frac{d}{dx}\left(\sum_{j=0}^{n} u_j\phi_j\right)\frac{d\phi_i}{dx}dx + c\int_0^1 \frac{d}{dx}\left(\sum_{j=0}^{n} u_j\phi_j\right)\phi_i dx = 0, \;\; i = 1, 2, ..., n-1. \qquad (18)
$$

Now the element stiffness matrix is given by,

$$S_e = \epsilon \begin{bmatrix} \int_{x_{i-1}}^{x_i} \frac{d\phi_{i-1}}{dx}\frac{d\phi_{i-1}}{dx}dx & \int_{x_{i-1}}^{x_i} \frac{d\phi_i}{dx}\frac{d\phi_{i-1}}{dx}dx \\ \int_{x_{i-1}}^{x_i} \frac{d\phi_{i-1}}{dx}\frac{d\phi_i}{dx}dx & \int_{x_{i-1}}^{x_i} \frac{d\phi_i}{dx}\frac{d\phi_i}{dx}dx \end{bmatrix} \tag{19}$$

$$+c \begin{bmatrix} \int_{x_{i-1}}^{x_i} \frac{d\phi_{i-1}}{dx}\phi_{i-1}dx & \int_{x_{i-1}}^{x_i} \frac{d\phi_i}{dx}\phi_{i-1}dx \\ \int_{x_{i-1}}^{x_i} \frac{d\phi_{i-1}}{dx}\phi_i dx & \int_{x_{i-1}}^{x_i} \frac{d\phi_i}{dx}\phi_i dx \end{bmatrix},$$

or,

$$S_e = \frac{\epsilon}{h}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} + \frac{c}{2}\begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix}, \tag{20}$$

where h is the length of an element. We have divided our domain uniformly, therefore h is a constant. The first element of the vector $\underline{b}$ is given by,

$$b_1 = -\epsilon \int_0^1 u_0 \frac{d\phi_0}{dx}\frac{d\phi_1}{dx}dx - c\int_0^1 u_0 \frac{d\phi_0}{dx}\phi_1 dx,$$

or,

$$b_1 = \frac{\epsilon}{h}u_0 + \frac{c}{2}u_0.$$

Similarly, $b_{n-1} = \frac{\epsilon}{h}u_n - \frac{c}{2}u_n$. All other elements of $\underline{b}$ are zero. Equation (18) in the matrix form is written as,

$$S\underline{u} = \underline{b},$$

or,

$$\underline{u} = S^{-1}\underline{b} \tag{21}$$

This equation has been implemented in MATLAB. Two graphs are shown in Figure 4. In the blue graph, $\epsilon = 1,\ c = 1$, while in the dotted graph, $\epsilon = 0.1,\ c = 4$. When convection dominates over diffusion term, we observe wiggles in the output, where the Standard Galerkin Approximation is used. To get rid of these undesired wiggles, we used a method called Streamline Upwind Petrov-Galerkin (SUPG).

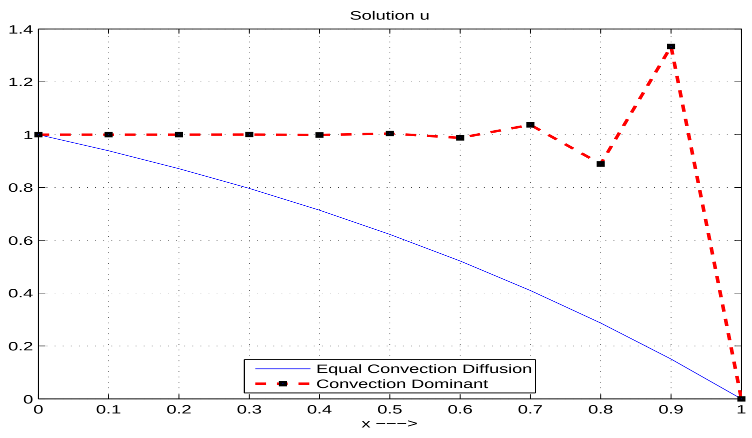

FIGURE 4. The numerical solution of the convection-diffusion equation, (blue) $\epsilon = 1,\ c = 1,\ (red)\ \epsilon = 0.1,\ c = 4$.

Streamline Upwind Petrov-Galerkin. In this method, a modified test function is used,

$$w(x) = v(x) + p(x),$$

where $v(x)$ is the classical test function. The additional term $p(x)$ introduces artificial diffusion and hence counters the convection dominance. This idea is taken from the finite difference method [1].

$$p(x) = \frac{h}{2}\frac{d\phi_i}{dx}\xi.$$

For a first order upwind, we have $\xi = 1$, which we will use in this study (other values of ξ are also used for different reasons). Hence,

$$w(x) = \phi_i + \frac{h}{2}\frac{d\phi_i}{dx}.$$

Using $w(x)$ in equation (17) instead of $v(x)$ we have,

$$\epsilon \int_0^1 \frac{du}{dx}\frac{dw}{dx}dx + c\int_0^1 \frac{du}{dx}w(x)dx = 0, \tag{22}$$

or,

$$\epsilon \int_0^1 \frac{d}{dx}\left(\sum_{j=0}^{n} u_j\phi_j\right)\frac{d}{dx}\left(\phi_i + \frac{h}{2}\frac{d\phi_i}{dx}\right)dx + c\int_0^1 \frac{d}{dx}\left(\sum_{j=0}^{n} u_j\phi_j\right)\left(\phi_i + \frac{h}{2}\frac{d\phi_i}{dx}\right)dx = 0. \tag{23}$$

The additional term of $\frac{h}{2}\frac{d\phi_i}{dx}$ is effective only in the convection part since elementwisely, the derivative of $p(x)$ is zero. The element matrix in this case is given by,

$$S_e = \epsilon\begin{bmatrix} \int_{x_{i-1}}^{x_i} \frac{d\phi_{i-1}}{dx}\frac{d\phi_{i-1}}{dx}dx & \int_{x_{i-1}}^{x_i} \frac{d\phi_i}{dx}\frac{d\phi_{i-1}}{dx}dx \\ \int_{x_{i-1}}^{x_i} \frac{d\phi_{i-1}}{dx}\frac{d\phi_i}{dx}dx & \int_{x_{i-1}}^{x_i} \frac{d\phi_i}{dx}\frac{d\phi_i}{dx}dx \end{bmatrix} + c\begin{bmatrix} \int_{x_{i-1}}^{x_i} \frac{d\phi_{i-1}}{dx}\phi_{i-1}dx & \int_{x_{i-1}}^{x_i} \frac{d\phi_i}{dx}\phi_{i-1}dx \\ \int_{x_{i-1}}^{x_i} \frac{d\phi_{i-1}}{dx}\phi_i dx & \int_{x_{i-1}}^{x_i} \frac{d\phi_i}{dx}\phi_i dx \end{bmatrix}$$
$$+\frac{hc}{2}\begin{bmatrix} \int_{x_{i-1}}^{x_i} \frac{d\phi_{i-1}}{dx}\frac{d\phi_{i-1}}{dx}dx & \int_{x_{i-1}}^{x_i} \frac{d\phi_i}{dx}\frac{d\phi_{i-1}}{dx}dx \\ \int_{x_{i-1}}^{x_i} \frac{d\phi_{i-1}}{dx}\frac{d\phi_i}{dx}dx & \int_{x_{i-1}}^{x_i} \frac{d\phi_i}{dx}\frac{d\phi_i}{dx}dx \end{bmatrix},$$

or,

$$S_e = \frac{\epsilon}{h}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} + \frac{c}{2}\begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix} + \frac{c}{2}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}. \tag{24}$$

b_1 takes the form,

$$b_1 = -\epsilon\int_0^1 u_0\frac{d\phi_0}{dx}\frac{d\phi_1}{dx}dx - c\int_0^1 u_0\frac{d\phi_0}{dx}\phi_1 dx - c\frac{h}{2}\int_0^1 u_0\frac{d\phi_0}{dx}\frac{d\phi_1}{dx}dx,$$

or,

$$b_1 = \frac{\epsilon}{h}u_0 + \frac{c}{2}u_0 + \frac{c}{2}u_0.$$

Note that the discretization of $u(x,t)$ is not changed. Now S and $\underline{b}$ are determined as usual and the equation (21) is used for the numerical solution. The solution obtained by the Streamline Upwind Petrov Galerkin (SUPG) method is shown in Figure 5, along with the exact solution, $u(x) = \frac{1}{1-e^m}\left(e^{mx} - e^m\right),\ m = \frac{c}{\epsilon}$. Again, we used $\epsilon = 0.1$, $c = 4$, but with the test function $w(x)$. We observe that the numerical solution in this case is smooth but it deviates from the analytical solution, where $u(x)$ varies sharply. This clipping effect is due to the artificial diffusion term and it can be reduced by taking a finer mesh. The result shown in Figure 6 indicates this fact. We expect that $\underline{u} \to u$ as $n \to \infty$. Again, a finer mesh means that extra memory and more computational time is required. For the current problem, a finer mesh also eliminates the need of SUPG method.

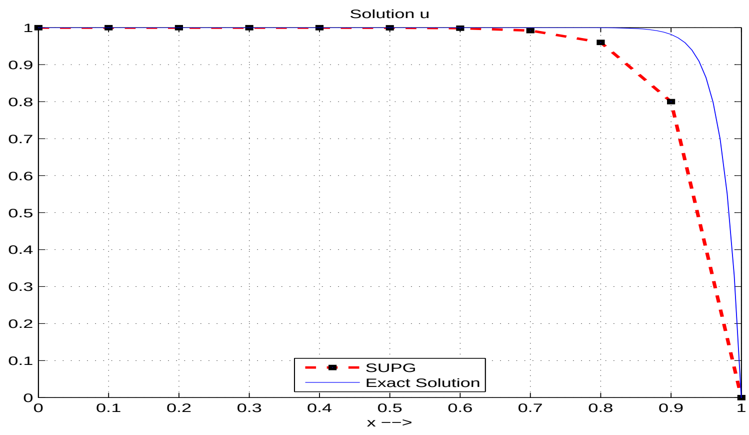

FIGURE 5. The solution of the convection-diffusion equation, $\epsilon = 0.1,\ c = 4,\ u(0) = 1,\ u(1) = 0$, (1) by SUPG method with $h = 0.1$ and (2) analytically, $u(x) = \frac{1}{1-e^m}\left(e^{mx} - e^m\right),\ m = \frac{c}{\epsilon},$.

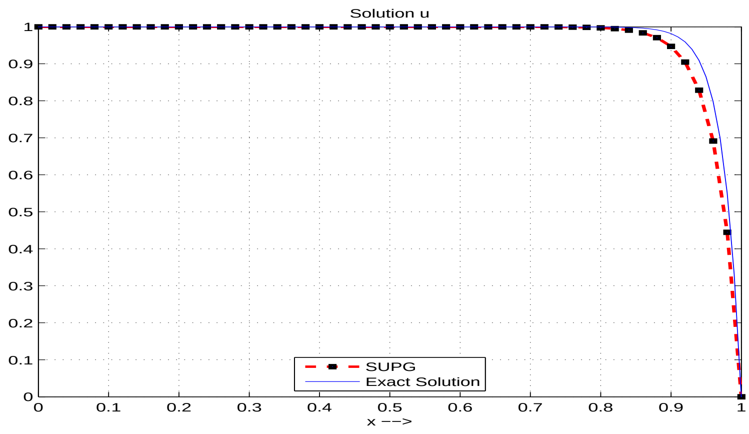

FIGURE 6. The solution of the convection-diffusion equation, $\epsilon = 0.1,\ c = 4,\ u(0) = 1,\ u(1) = 0$, (1) by SUPG method with $h = 0.02$ and (2) analytically, $u(x) = \frac{1}{1-e^m}\left(e^{mx} - e^m\right),\ m = \frac{c}{\epsilon},$.

3.1. **Solution of Transient Convection Diffusion Equation.**

Consider the following time-dependent convection-diffusion equation,

$$\frac{\partial u(x,t)}{\partial t} + c\frac{\partial u(x,t)}{\partial x} = D\frac{\partial^2 u(x,t)}{\partial x^2}. \tag{25}$$

$$u(0,t) = 1,\ u(1,t) = (2-t),\ u(x,0) = \frac{1}{e-1}(e^x - 1) + 1.$$

Here D and c, represent the diffusion coefficient and the flow velocity and these quantities are assumed to be constant. We can use the Newton-Cotes rule in case c and D are specified by some given functions of x. Initial conditions $u(x,0)$ and boundary values $u(0,t)$ and $u(1,t)$ are also given. After multiplying equation (25) by a test function $v(x)$ of our choice and integrating over the problem domain, we get the weak formulation.

$$\int_0^1 \frac{\partial u(x,t)}{\partial t} v(x)dx + c\int_0^1 \frac{\partial u(x,t)}{\partial x} v(x)dx = -D\int_0^1 \frac{\partial u(x,t)}{\partial x}\frac{\partial v(x)}{\partial x}dx. \tag{26}$$

We divide our domain into n elements as explained in the previous section, then the discretization of $u(x,t)$ is done as follows

$$u(x,t) \approx \sum_{j=0}^{n} u_j(t)\phi_j(x), \text{ where } u_0(t) \text{ and } u_n(t) \text{ are given.}$$

For brevity, we will use the notation $u_j(t) = u_j$ and $\phi_j(x) = \phi_j$.

$$\int_0^1 \frac{\partial}{\partial t}\sum_{j=0}^{n} u_j\phi_j\phi_i dx + c\int_0^1 \frac{\partial}{\partial x}(\sum_{j=0}^{n} u_j\phi_j)\phi_i dx = -D\int_0^1 \frac{\partial}{\partial x}(\sum_{j=0}^{n} u_j\phi_j)\frac{\partial \phi_i}{\partial x}dx, \quad i = 1, 2, .., n-1. \tag{27}$$

or,

$$\frac{d}{dt}\sum_{j=1}^{n-1} u_j \int_0^1 \phi_j\phi_i dx = -c\sum_{j=1}^{n-1} u_j\int_0^1 \frac{d\phi_j}{dx}\phi_i dx - D\sum_{j=1}^{n-1} u_j\int_0^1 \frac{d\phi_j}{dx}\frac{d\phi_i}{dx}dx + b_i, \tag{28}$$

where,

$$b_i = \begin{cases} -u_0\int_{x_0}^{x_1}\left(D\frac{d\phi_0}{dx}\frac{d\phi_1}{dx} + c\frac{d\phi_0}{dx}\phi_1\right)dx, & \text{for } i = 1, \\ -u_n\int_{x_{n-1}}^{x_n}\left(D\frac{d\phi_n}{dx}\frac{d\phi_{n-1}}{dx} + c\frac{d\phi_n}{dx}\phi_{n-1}\right)dx, & \text{for } i = n-1, \\ 0, & 1 < i < n-1, \end{cases}$$

or,

$$b_i = \begin{cases} u_0\frac{D}{h} + u_0\frac{c}{2} & \text{for } i = 1, \\ u_n\frac{D}{h} - u_n\frac{c}{2}, & \text{for } i = n-1, \\ 0, & 1 < i < n-1. \end{cases}$$

The matrix form of the equation(28) is given by,

$$M\frac{d\underline{u}}{dt} = -S\underline{u} + \underline{b}, \tag{29}$$

$$\tag{30}$$

where the $(i,j)th$ element of the mass matrix M is,

$$M_{i,j} = \int_0^1 \phi_i\phi_j dx.$$

Its element matrix M_e is given by,

$$M_e = D\begin{bmatrix} \int_{x_{i-1}}^{x_i}\phi_{i-1}(x)\phi_{i-1}(x)dx & \int_{x_{i-1}}^{x_i}\phi_i(x)\phi_{i-1}(x)dx \\ \int_{x_{i-1}}^{x_i}\phi_{i-1}(x)\phi_i(x)dx & \int_{x_{i-1}}^{x_i}\phi_i(x)\phi_i(x)dx \end{bmatrix}, \tag{31}$$

or,

$$M_e = \frac{h}{6}\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}. \tag{32}$$

The procedure for the assembly of the matrix M is exactly the same as for the matrix S. Applying the the Forward Euler scheme for the time discretization, we have,

$$M\frac{\underline{u}^{p+1} - \underline{u}^p}{\Delta t} = -S\underline{u}^p + \underline{b}^p, \tag{33}$$

where p is the time index. We have taken a time step $\Delta t = 0.01$, hence $p = 1, 2, 3...$ corresponds to $\Delta t, 2\Delta t, 3\Delta t...$, in the time scale. Equation (33) can be written as,

$$\underline{u}^{p+1} = (I - \Delta t M^{-1} S)\underline{u}^{p} + \Delta t M^{-1}\underline{b}^{p}. \tag{34}$$

Note that $\underline{u}^{p}$ is already computed in the previous iteration p. This time-discretization scheme is called "explicit". In the explicit scheme we are faced with a stability condition. Let λ be an eigenvalue of $I - \Delta t M^{-1} S$, then Δt should be chosen such that,

$$|\lambda| \leq 1, \ \forall\ \lambda.$$

Otherwise we will have an unstable result. For the current example we have, $h = 0.1,\ \Delta t = 0.01,\ n = 10,\ D = 1,\ u(0, t) = 0,\ u(1, t) = (1-t),\ u(x, 0) = \frac{e}{e-1}(1-e^{-x})$. The result is shown in Figure 7.

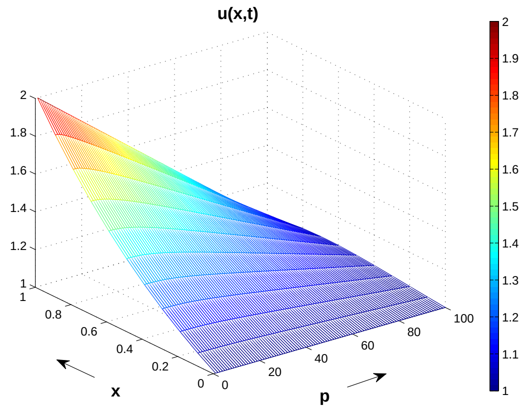

FIGURE 7. The numerical solution of the time dependent convection-diffusion equation by SGA, $D = 1,\ c = 1,\ u(0, t) = 1,\ u(1, t) = (2 - t),\ u(x, 0) = \frac{1}{e-1}(e^{x} - 1) + 1$

Implicit Scheme: If we use Backward Euler scheme for equation (30), we have

$$M\frac{\underline{u}^{p+1} - \underline{u}^{p}}{\Delta t} = -S\underline{u}^{p+1} + \underline{b}^{p+1}.$$

Let $G = (M + \Delta t S)^{-1}$, then,

$$\underline{u}^{p+1} = GM\underline{u}^{p} + \Delta t G\underline{b}^{p+1}.$$

This equation is implemented in MATLAB. The result is not shown because it is visually the same as for the explicit case. An advantage of the implicit method is that it is unconditionally stable. But, in general, it is relatively more demanding in terms of use of memory and processing time.

3.2. **Solution of the Non-Linear Burgers' Equation by Finite Elements.** Consider the following non-linear equation, also called Burgers' Equation,

$$\frac{\partial u(x,t)}{\partial t} + u(x,t)\frac{\partial u(x,t)}{\partial x} = D\frac{\partial^2 u(x,t)}{\partial x^2}, \tag{35}$$

$$u(0, t) = 1,\ u(1, t) = (2 - t),\ u(x, 0) = \frac{e}{e-1}(1 - e^{-x}) + 1.$$

The weak formulation of equation (35) is given by,

$$\int_0^1 \frac{\partial u(x,t)}{\partial t}v(x)dx + \int_0^1 u(x,t)\frac{\partial u(x,t)}{\partial x}v(x)dx = -D\int_0^1 \frac{\partial u(x,t)}{\partial x}\frac{\partial v(x,t)}{\partial x}dx. \tag{36}$$

In this equation the non-linear term $u\frac{\partial u}{\partial x}$ is the difficult part. It can be treated by a number of ways. One option is that we use the same discretization for every u, given by,

$$u(x,t) \approx \sum_{j=0}^{n} u_j(t)\phi_j(x), \tag{37}$$

where $u_0(t)$ and $u_n(t)$ are known. With this scheme, the given non-linear equation takes the form,

$$\int_0^1 \frac{\partial}{\partial t}\sum_{j=0}^{n} u_j\phi_j\phi_i dx + \int_0^1 \sum_{j=0}^{n} u_j\phi_j \left(\frac{\partial}{\partial x}\sum_{k=0}^{n} u_k\phi_k\right)\phi_i dx$$
$$= -D\int_0^1 \frac{\partial}{\partial x}(\sum_{j=0}^{n} u_j\phi_j)\frac{\partial\phi_i}{\partial x}dx,\ i = 1,2,..,n-1, \tag{38}$$

or,

$$\frac{d}{dt}\sum_{j=1}^{n-1} u_j\int_0^1 \phi_j\phi_i dx = -D\sum_{j=1}^{n-1} u_j\int_0^1 \frac{d\phi_j}{dx}\frac{d\phi_i}{dx}dx - L_i + b_i,\ i = 1,2,...,n-1, \tag{39}$$

where L_i results from the non-linear term and b_i contains the boundary values. Computation of L_i, i.e., the discretized form of the non-linear term in the ith row is given by,

$$L_i = \int_0^1 \left(\sum_{k=0}^{n} u_k\phi_k\right)\left(\frac{\partial}{\partial x}\sum_{j=0}^{n} u_j\phi_j\right)\phi_i dx,\ i = 2,3..,n-2. \tag{40}$$

We will determine L_1 and L_{n-1} separately, because they are slightly different from the general form L_i. Now considering only the nonzero terms in equation (40), we have,

$$L_i = \int_{x_{i-1}}^{x_i} (u_{i-1}\phi_{i-1} + u_i\phi_i)\left(u_{i-1}\frac{d\phi_{i-1}}{dx} + u_i\frac{d\phi_i}{dx}\right)\phi_i dx +$$
$$\int_{x_i}^{x_{i+1}} (u_i\phi_i + u_{i+1}\phi_{i+1})\left(u_i\frac{d\phi_i}{dx} + u_{i+1}\frac{d\phi_{i+1}}{dx}\right)\phi_i dx,$$

$$L_i = \frac{1}{h}(-u_{i-1} + u_i)\int_{x_{i-1}}^{x_i} (u_{i-1}\phi_{i-1} + u_i\phi_i)\phi_i dx +$$
$$\frac{1}{h}(-u_i + u_{i+1})\int_{x_i}^{x_{i+1}} (u_i\phi_i + u_{i+1}\phi_{i+1})\phi_i dx,$$

$$L_i = \frac{1}{h}(-u_{i-1} + u_i)\left(u_{i-1}\frac{h}{6} + u_i\frac{h}{3}\right) + \frac{1}{h}(-u_i + u_{i+1})\left(u_i\frac{h}{3} + u_{i+1}\frac{h}{6}\right),$$

$$L_i = \frac{1}{6}[-(u_{i-1})^2 - u_{i-1}u_i + u_iu_{i+1} + (u_{i+1})^2].$$

Now L_1 and L_{n-1} are given by,

$$L_1 = \frac{1}{6}[-u_0u_1 + u_1u_2 + (u_2)^2],$$

$$L_{n-1} = \frac{1}{6}[-(u_{n-2})^2 - u_{n-2}u_{n-1} + u_{n-1}u_n].$$

In L_1 and L_{n-1}, we did not consider $\frac{-1}{6}u_0^2$ and $\frac{1}{6}u_n^2$. These known values are taken by b_1 and b_{n-1}, respectively,

$$b_1 = \frac{(u_0)^2}{6} + u_0\frac{D}{h},$$

$$b_{n-1} = -\frac{(u_n)^2}{6} + u_n\frac{D}{h}.$$

The matrix form of equation (39) is given by,

$$M\frac{d\underline{u}}{dt} = -S\underline{u} - \underline{L} + \underline{b}.$$

After applying the Euler Backward scheme, we get,

$$(M + \Delta t S)\underline{u}^{p+1} = -\Delta t\underline{L}^{p+1} + M\underline{u}^p + \Delta t\underline{b}^{p+1},$$

where p is a time index and $\underline{L}^{p+1} = \underline{L}(\underline{u}^{p+1})$. In a Picard iteration, we already have u^{p+1}, either as an initial guess or a value close to the final solution, in case we have convergence. Let $G = (M + \Delta t S)^{-1}$, then,

$$\underline{u}^{p+1} = G(-\Delta t\underline{L}^{p+1} + M\underline{u}^p + \Delta t\underline{b}^{p+1}).$$

Let $\underline{b}1 = G(M\underline{u}^p + \Delta t\underline{b}^{p+1})$, which is a constant for each time iteration p.

$$\underline{u}^{p+1} = -\Delta t G\underline{L}^{p+1} + \underline{b}1. \tag{41}$$

This equation is solved by Picard Iteration. Picard iteration is used to solve non-linear equations written in the form,

$$\underline{u}_{new} = f(\underline{u}_{old}). \tag{42}$$

Where $\underline{u}_{old}$ starts with an initial guess. We calculate $\underline{u}_{new}$ by using equation (42). In the next iteration, the value of $\underline{u}_{new}$ is assigned to $\underline{u}_{old}$. This procedure goes on until a stopping condition is satisfied. The iteration stops successfully if the infinity norm of $\frac{\underline{u}_{new}-\underline{u}_{old}}{\underline{u}_{new}}$ is less than a specified value, i.e.,

$$\max\left(\left|\frac{\underline{u}_{new} - \underline{u}_{old}}{\underline{u}_{new}}\right|\right) < \delta,$$

where δ is an acceptable error level. In this case we move on to the next time step. If this condition is not met within a specified number of iterations or $\underline{u}_{new} \to \infty$, the Picard iteration stops, unsuccessfully. Equation (42) can be formulated in many ways. If it is divergent in one form, it might converge in another form. For the current time dependent problem, we used $\underline{u}^p$ as the initial guess in Picard iteration for $\underline{u}^{p+1}$. This procedure is more elaborated in Algorithm 1. The approximate solution $u(x_j, p)$ is shown in Figure 8. In Algorithm 1, it can be seen that the vector $\underline{L}$ is computed from $\underline{u}^{p+1}$, but $\underline{u}^{p+1}$ is taken from the previous Picard iteration. Hence the entire non-linear term is treated at the previous Picard iteration.

Semi-Implicit and Fully Implicit Methods

We reconsider the non-linear problem (equation (35)), first discretize it in time and then in space. In this way we take certain unknown variables at previous time iteration p (the semi-implicit scheme), hence the problem is linearized.

$$\frac{u^{p+1} - u^p}{\Delta t} + u^p\frac{\partial u^{p+1}}{\partial x} = D\frac{\partial^2 u^{p+1}}{\partial x^2},\ x \in (0,1),\ p = 1, 2, 3, ..., \tag{43}$$

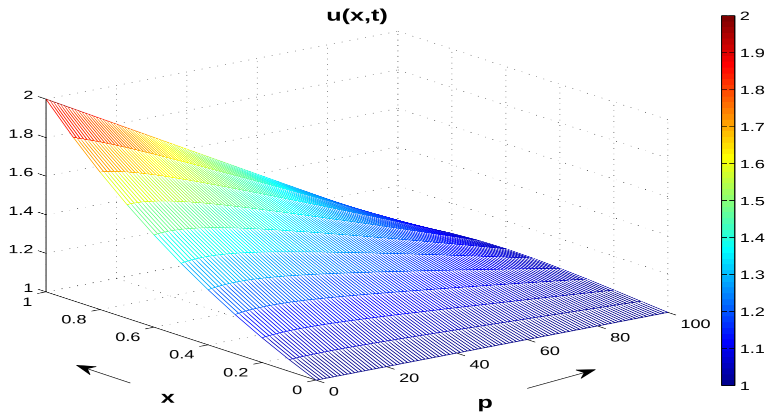

FIGURE 8. The numerical solution of Burgers' equation by SGA, $D = 1,\ u(0,t) = 1,\ u(1,t) = (2-t),\ u(x,0) = \frac{1}{e-1}(e^x - 1) + 1$

Solution Algorithm for the Non-linear Burgers' Equation

```
$n$=number of elements,  $h = \frac{1}{n},\ x_j = jh,\ j = 0, 1, ..., n,$
$u_0(t) = 0,\ u_n(t) = (1-t),\ u(x,0) = \frac{1}{e-1}(e^x - 1),$
Initialize $\underline{L}$, $\underline{b}$, $M$, and $S$ with zeros,
Compute $Me$, $Se$, $M$, $S$, and $G$,

Time Loop Starts with p=1,
    $u^{p+1} = u^p$, %serves as guess
    Compute $b_1,\ b_{n-1},\ \underline{b}1,$

    Picard Loop starts here,
        Compute $L$,
        $y = -\Delta t G L + \underline{b}1$
        If stopping condition=true, break Picard Loop
        $u^{p+1} = y$
    Picard Loop Ends
    If condition=divergence, break Time Loop
Time Loop Ends
Display results
```

where $u^p = u(x, p\Delta t)$. The weak form and the corresponding space-discretization are given by,

$$\frac{1}{\Delta t}\int_0^1 \left(u^{p+1} - u^p\right) v(x)dx + \int_0^1 u^p(x,t)\frac{\partial u^{p+1}}{\partial x}v(x)dx = -D\int_0^1 \frac{\partial u^{p+1}}{\partial x}\frac{dv}{dx}dx, \tag{44}$$

$$\frac{1}{\Delta t}\int_0^1 \sum_{j=0}^{n}\left(u_j^{p+1} - u_j^p\right)\phi_j\phi_i(x)dx + \int_0^1 \left(\sum_{k=0}^{n} u_k^p\phi_k\right)\left(\frac{\partial}{\partial x}\sum_{j=0}^{n} u_j^{p+1}\phi_j\right)\phi_i dx$$
$$+D\int_0^1 \frac{\partial}{\partial x}\left(\sum_{j=0}^{n} u_j^{p+1}\phi_j\right)\frac{d\phi_i}{dx}dx = 0,\ i = 1, 2, ..., n-1. \tag{45}$$

In the following computations, we used a Newton-Cotes integration rule (the Trapezoidal rule given by equation (13)). The element stiffness matrix is determined in the following way,

$$S_{e_{1,1}} = \int_{x_{i-1}}^{x_i} \left(u_{i-1}^p \phi_{i-1} + u_i^p \phi_i\right) \frac{d\phi_{i-1}}{dx} \phi_{i-1} dx + D \int_{x_{i-1}}^{x_i} \frac{d\phi_{i-1}}{dx} \frac{d\phi_{i-1}}{dx} dx,$$

$$S_{e_{1,1}} = -\frac{u_{i-1}^p}{2} + \frac{D}{h}.$$

The complete element matrix is given by,

$$S_e = \frac{D}{h} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} + \frac{1}{2} \begin{bmatrix} -u_{i-1}^p & u_{i-1}^p \\ -u_i^p & u_i^p \end{bmatrix}. \tag{46}$$

Since, S is a function of u^p, therefore, S_e and S are computed at each time iteration. We will denote S by S^p. The mass element matrix M_e is different in this case, because we have not computed it analytically.

$$M_e = \frac{h}{2} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{47}$$

The mass matrix M is assembled as usual. The first element of the vector $\underline{b}^p$ is given by,

$$b_1^p = -\int_0^1 \left(u_0^p \phi_0 + u_1^p \phi_1\right) u_0^{p+1} \frac{d\phi_0}{dx} \phi_1 dx - D \int_0^1 u_0^{p+1} \frac{d\phi_0}{dx} \frac{d\phi_1}{dx} dx,$$

or,

$$b_1^p = \frac{u_0^{p+1} u_1^p}{2} + \frac{u_0^{p+1} D}{h}.$$

Similarly,

$$b_{n-1}^p = -\frac{u_n^{p+1} u_{n-1}^p}{2} + \frac{u_n^{p+1} D}{h}.$$

Now the equation (45) written in a matrix form is given by,

$$M \frac{\underline{u}^{p+1} - \underline{u}^p}{\Delta t} + S^p \underline{u}^{p+1} = \underline{b}^p, \tag{48}$$

$$\underline{u}^{p+1} = \left(M + \Delta t S^p\right)^{-1} \left(M \underline{u}^p + \Delta t \underline{b}^p\right). \tag{49}$$

The numerical solution is shown in Figure 9. Now we consider the fully implicit

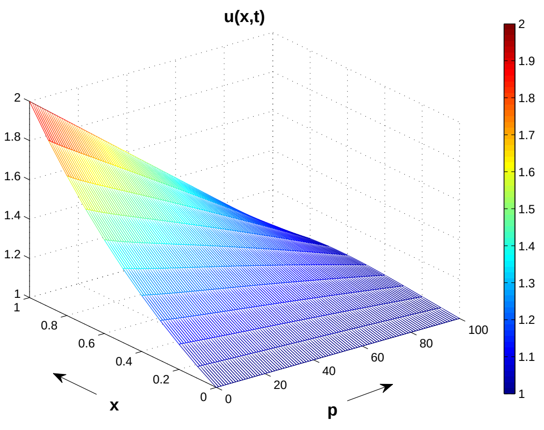

Figure 9. The numerical solution of Burgers' equation by the semi-implicit scheme, $D = 1$, $u(0,t) = 1$, $u(1,t) = (2 - t)$, $u(x,0) = \frac{1}{e-1}(e^x - 1) + 1$

case. In this case, again Euler Backward scheme is used for discretization of the time derivative but all other variables are taken at the current time iteration $p+1$. We used a Picard method to solve the non-linear problem, so it is linearized with respect to Picard iteration. The time-discretized form of equation (35) is given by,

$$\frac{u^{p+1}_{new} - u^p}{\Delta t} + u^{p+1}_{old}\frac{\partial u^{p+1}_{new}}{\partial x} = D\frac{\partial^2 u^{p+1}_{new}}{\partial x^2},\ x \in (0,1),\ p = 1,2,3,... \tag{50}$$

The method explained in Algorithm 3.2 is equivalent to the following linearization.

$$\frac{u^{p+1}_{new} - u^p}{\Delta t} + u^{p+1}_{old}\frac{\partial u^{p+1}_{old}}{\partial x} = D\frac{\partial^2 u^{p+1}_{new}}{\partial x^2},\ x \in (0,1),\ p = 1,2,3,... \tag{51}$$

The element stiffness matrix corresponding to equation (50) is given by,

$$S_e = \frac{D}{h}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} + \frac{1}{2}\begin{bmatrix} -(u^{p+1}_{i-1})_{old} & (u^{p+1}_{i-1})_{old} \\ -(u^{p+1}_{i})_{old} & (u^{p+1}_{i})_{old} \end{bmatrix}, \tag{52}$$

Similarly the matrix form corresponding to the discretization of equation (50) is given by,

$$M\frac{(\underline{u}^{p+1})_{new} - \underline{u}^p}{\Delta t} + (S^{p+1})_{old}(\underline{u}^{p+1})_{new} = \left(\underline{b}^{p+1}\right)_{old}, \tag{53}$$

$$(\underline{u}^{p+1})_{new} = \left(M + \Delta t (S^{p+1})_{old}\right)^{-1}\left(M\underline{u}^p + \Delta t\left(\underline{b}^{p+1}\right)_{old}\right). \tag{54}$$

Equation (54) is solved by Picard iteration. The result is close to the semi-implicit case. For the comparison, their difference is shown in Figure 10. The difference decreases with the increasing time.

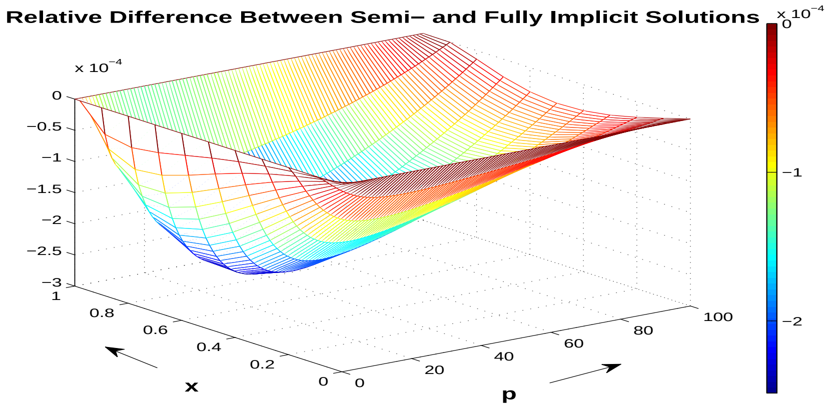

FIGURE 10. The difference in two solutions of Burgers' equation by Semi-Implicit and Fully-Implicit schemes, $D = 1,\ u(0,t) = 1,\ u(1,t) = (2-t),\ u(x,0) = \frac{1}{e-1}(e^x - 1)$

4. System of Non-linear Equations

In this section, we have computed a numerical solution for a system of nonlinear, coupled equations. The system of equations is given by,

$$\frac{\partial(\rho h)}{\partial t} + \frac{\partial(\rho h v)}{\partial x} = D_1\frac{\partial^2 T}{\partial x^2} + q(x,t), \tag{55}$$

$$\frac{\partial \rho}{\partial t} + \frac{\partial(\rho v)}{\partial x} = 0, \tag{56}$$

$$v(x,t) = -\lambda_1 \frac{\partial P}{\partial x}, \tag{57}$$

$$P(x,t).V = RT(x,t), \tag{58}$$

$$h(x,t) = cT(x,t). \tag{59}$$

This is a one-dimensional, simplified case of the flow of a certain fluid through a porous medium. The system is taken from Master Thesis of Abdelhaq Abouhafç. In this system, the unknown variables are the density $\rho(x,t)$, the velocity $v(x,t)$, the pressure P, the temperature T, and the enthalpy $h(x,t)$. On the other hand the volume V, the diffusion coefficient D_1, the universal gas constant R, λ_1, and c are given constants. The source function $q(x,t)$ is also known. In this section, the symbol h and v are already taken, so we will use the symbol Δx for the length of an element and $\eta(x)$ for the classical basis function. We have used both, the semi-implicit and the fully implicit scheme to solve this problem. After eliminating P and T, we rewrite the above system as,

$$\frac{\partial(\rho h)}{\partial t} + \frac{\partial(\rho h v)}{\partial x} = D\frac{\partial^2 h}{\partial x^2} + q(x,t), \tag{60}$$

$$\frac{\partial \rho}{\partial t} + \frac{\partial(\rho v)}{\partial x} = 0, \tag{61}$$

$$v(x,t) = -\lambda \frac{\partial h}{\partial x}, \tag{62}$$

where $D = D_1/c$ and $\lambda = \frac{\lambda_1 R}{cV}$. From equation (60) and (61) we have,

$$\rho\frac{\partial h}{\partial t} + h\frac{\partial \rho}{\partial t} + h\frac{\partial(\rho v)}{\partial x} + \rho v\frac{\partial h}{\partial x} = D\frac{\partial^2 h}{\partial x^2} + q(x,t), \tag{63}$$

$$\frac{\partial \rho}{\partial t} = -\frac{\partial(\rho v)}{\partial x}. \tag{64}$$

Using equation (64) into equation (63), we have,

$$\rho\frac{\partial h}{\partial t} - h\frac{\partial(\rho v)}{\partial x} + h\frac{\partial(\rho v)}{\partial x} + \rho v\frac{\partial h}{\partial x} = D\frac{\partial^2 h}{\partial x^2} + q(x,t), \tag{65}$$

or,

$$\rho\frac{\partial h}{\partial t} + \rho v\frac{\partial h}{\partial x} = D\frac{\partial^2 h}{\partial x^2} + q(x,t). \tag{66}$$

Using $v = -\lambda\frac{\partial h}{\partial x}$ in equations (66) and (61) we have the following system of equations, ready to be solved numerically.

$$\rho\frac{\partial h}{\partial t} - \lambda\rho\frac{\partial h}{\partial x}\frac{\partial h}{\partial x} = D\frac{\partial^2 h}{\partial x^2} + q(x,t), \tag{67}$$

$$\frac{\partial \rho}{\partial t} - \lambda\frac{\partial h}{\partial x}\frac{\partial \rho}{\partial x} - \lambda\rho\frac{\partial^2 h}{\partial x^2} = 0. \tag{68}$$

The initial conditions and the boundary values are given by,

$$h(x,0) = 3 - x,\ h(0,t) = 2 + sin5t,\ h(1,t) = 2 + cost$$

$$\rho(x,0) = 1.5 - x,\ \rho(0,t) = 1.5$$

These equations are linearized in the same manner as in case of Burgers' equation i.e., not more than one variable is taken at current time index, in each term (semi-implicit case). Equations (67) and (68) take the forms (with q(x,t)=0),

$$\rho^p\frac{h^{p+1} - h^p}{\Delta t} - \lambda\rho^p\frac{\partial h^p}{\partial x}\frac{\partial h^{p+1}}{\partial x} = D\frac{\partial^2 h^{p+1}}{\partial x^2}, \tag{69}$$

$$\frac{\rho^{p+1}-\rho^p}{\Delta t} - \lambda \frac{\partial h^p}{\partial x}\frac{\partial \rho^{p+1}}{\partial x} - \lambda \rho^{p+1}\frac{\partial^2 h^p}{\partial x^2} = 0. \tag{70}$$

Now the weak form of equation (69) and (70) is given by,

$$\frac{1}{\Delta t}\int_0^1 \rho^p \left(h^{p+1}-h^p\right)\eta(x)dx - \lambda \int_0^1 \rho^p \frac{\partial h^p}{\partial x}\frac{\partial h^{p+1}}{\partial x}\eta(x)dx = -D\int_0^1 \frac{\partial h^{p+1}}{\partial x}\frac{d\eta}{dx}dx, \tag{71}$$

$$\begin{aligned}\frac{1}{\Delta t}\int_0^1 \left(\rho^{p+1}-\rho^p\right)w(x)dx - \lambda \int_0^1 \frac{\partial h^p}{\partial x}\frac{\partial \rho^{p+1}}{\partial x}w(x)dx \\ -\lambda \int_0^1 \rho^{p+1}\frac{\partial}{\partial x}\left(\frac{\partial h^p}{\partial x}\right)w(x)dx = 0.\end{aligned} \tag{72}$$

We have used the following discretization for unknown variables,

$$h(x,t) \approx \sum_{i=0}^{n} h_i(t)\phi_i(x), \;\; \rho(x,t) \approx \sum_{i=0}^{n} \rho_i(t)\phi_i(x).$$

Now the discretization of equation (71) is given by,

$$\begin{aligned}\frac{1}{\Delta t}\int_0^1 \sum_{k=0}^{n}\rho_k^p\phi_k \sum_{j=0}^{n}(h_j^{p+1}-h_j^p)\phi_j\phi_i dx - \lambda\int_0^1 \sum_{k=0}^{n}\rho_k^p\phi_k \sum_{l=0}^{n} h_l^p\frac{\partial \phi_l}{\partial x}\sum_{j=0}^{n} h_j^{p+1}\frac{\partial \phi_j}{\partial x}\phi_i dx \\ = -D\int_0^1 \sum_{j=0}^{n} h_j^{p+1}\frac{\partial \phi_j}{\partial x}\frac{d\phi_i}{dx}dx \qquad i = 1, 2..., n-1.\end{aligned} \tag{73}$$

We will use this equation to find the enthalpy h. Therefore we add an h subscript to the matrices corresponding to this equation. The first element of the mass element matrix M_{h_e}, is given by,

$$M_{h_{e_{1,1}}} = \int_{x_{i-1}}^{x_i} \left(\rho_{i-1}^p\phi_{i-1} + \rho_i^p\phi_i\right)\phi_{i-1}\phi_{i-1}dx = \frac{\Delta x}{2}\rho_{i-1}^p, \tag{74}$$

where we have used a Newton-Cotes integration rule (the trapezoidal rule). The complete element matrix M_{h_e} is given by,

$$M_{h_e} = \frac{\Delta x}{2}\begin{bmatrix} \rho_{i-1}^p & 0 \\ 0 & \rho_i^p \end{bmatrix}. \tag{75}$$

Similarly, the first element in the stiffness element matrix S_{h_e}, is determined as follows,

$$\begin{aligned}S_{h_{e_{1,1}}} = -\lambda\int_{x_{i-1}}^{x_i} \left(\rho_{i-1}^p\phi_{i-1} + \rho_i^p\phi_i\right)\left(\rho_{i-1}^p\frac{d\phi_{i-1}}{dx} + \rho_i^p\frac{d\phi_i}{dx}\right)\frac{d\phi_{i-1}}{dx}\phi_{i-1}dx \\ -D\int_0^1 \frac{\partial \phi_{i-1}}{\partial x}\frac{d\phi_{i-1}}{dx}dx,\end{aligned} \tag{76}$$

$$S_{h_{e_{1,1}}} = \frac{\lambda}{2\Delta x}\left(-h_{i-1}^p + h_i^p\right)\rho_{i-1}^p + \frac{D}{\Delta x}. \tag{77}$$

The complete element matrix S_{h_e} is given by,

$$S_{h_e} = \frac{\lambda}{2\Delta x}\left(-h_{i-1}^p + h_i^p\right)\begin{bmatrix} \rho_{i-1}^p & -\rho_{i-1}^p \\ \rho_i^p & -\rho_i^p \end{bmatrix} + \frac{D}{\Delta x}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}. \tag{78}$$

From these element matrices, we assemble global matrices, M_h and S_h. Let b_h be a vector, which includes boundary values corresponding to variables in equation (69).

Then $b_{h_1} = -S_{h_{2,1}} h_0$ and $b_{h_{n-1}} = -S_{h_{n-1,n}} h_n$. The element matrices corresponding to equation (72) are given as follows (Streamline Upwind Petrov-Galerkin applied),

$$M_{\rho_e} = \frac{\Delta x}{4} \begin{bmatrix} 1 & -1 \\ 1 & 3 \end{bmatrix}, \tag{79}$$

$$S_{\rho_e} = \frac{\lambda \left(-h^p_{i-1} + h^p_i\right)}{2\Delta x} \begin{bmatrix} 0 & 0 \\ 2 & -2 \end{bmatrix}. \tag{80}$$

The final equations in the matrix form are given by,

$$\underline{h}^{p+1} = \left(M^p_h + \Delta t S^p_h\right)^{-1} \left(M^p_h \underline{h}^p + \Delta t \underline{b}^p_h\right),$$

$$\underline{\rho}^{p+1} = \left(M_\rho + \Delta t S^p_\rho\right)^{-1} \left(M_\rho \underline{\rho}^p + \Delta t \underline{b}^p_\rho\right).$$

The numerical solution obtained from these equations is shown in Figure 11 for two time resolutions $\Delta t = 0.01$ and $\Delta t = 0.1$. The remaining variables, v, P, and T are computed as post processing. In case of fully implicit scheme, equation (67) and

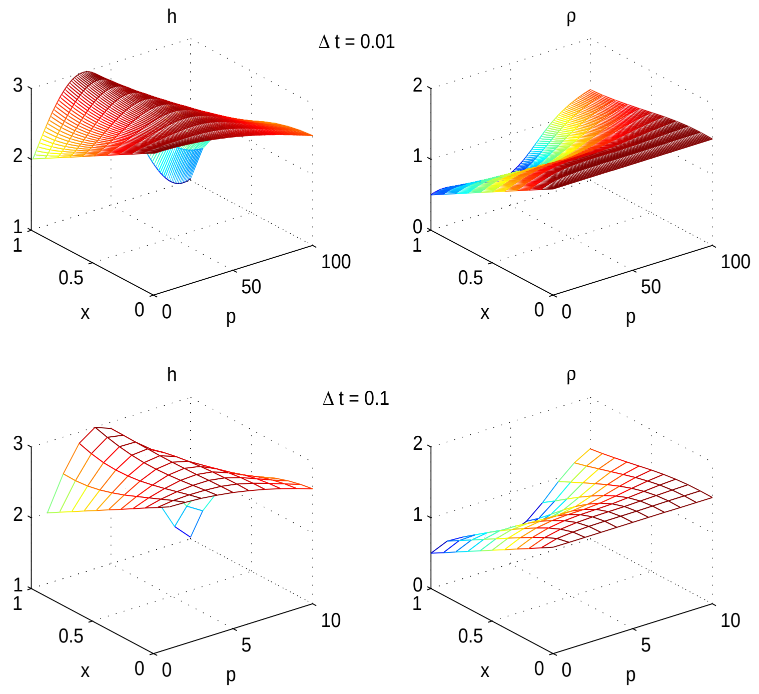

FIGURE 11. The numerical solution of h and ρ, $D = 1, q = 0, \Delta x = 0.1,\ h(x,0) = 3 - x,\ h(0,t) = 2 + sin5t,\ h(1,t) = 2 + cost\rho(x,0) = 1.5 - x,\ \rho(0,t) = 1.5$

(68) are linearized in the following way (q=0),

$$\left(\rho^{p+1}\right)_{old} \frac{\left(h^{p+1}\right)_{new} - h^p}{\Delta t} - \lambda \left(\rho^{p+1}\right)_{old} \frac{\partial \left(h^{p+1}\right)_{old}}{\partial x} \frac{\partial \left(h^{p+1}\right)_{new}}{\partial x} = D \frac{\partial^2 \left(h^{p+1}\right)_{new}}{\partial x^2}, \tag{81}$$

$$\frac{\left(\rho^{p+1}\right)_{new} - \rho^p}{\Delta t} - \lambda \frac{\partial \left(h^{p+1}\right)_{old}}{\partial x} \frac{\partial \left(\rho^{p+1}\right)_{new}}{\partial x} - \lambda \left(\rho^{p+1}\right)_{new} \frac{\partial^2 \left(h^{p+1}\right)_{old}}{\partial x^2} = 0. \tag{82}$$

Where the subscript 'old' indicates the value at previous Picard iteration. The element matrices are given by,

$$M_{h_e} = \frac{\Delta x}{2} \begin{bmatrix} \left(\rho_{i-1}^{p+1}\right)_{old} & 0 \\ 0 & \left(\rho_{i}^{p+1}\right)_{old} \end{bmatrix}. \tag{83}$$

$$S_{h_e} = \frac{\lambda}{2\Delta x}\left(-\left(h_{i-1}^{p+1}\right)_{old} + \left(h_{i}^{p+1}\right)_{old}\right) \begin{bmatrix} \left(\rho_{i-1}^{p+1}\right)_{old} & -\left(\rho_{i-1}^{p+1}\right)_{old} \\ \left(\rho_{i}^{p+1}\right)_{old} & -\left(\rho_{i}^{p+1}\right)_{old} \end{bmatrix} + \frac{D}{\Delta x}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}. \tag{84}$$

$$S_{\rho_e} = \frac{\lambda\left(-\left(h_{i-1}^{p+1}\right)_{old} + \left(h_{i}^{p+1}\right)_{old}\right)}{2\Delta x} \begin{bmatrix} 0 & 0 \\ 2 & -2 \end{bmatrix}. \tag{85}$$

The final equations in the matrix from are given by,

$$\left(\underline{h}^{p+1}\right)_{new} = \left(\left(M_h^{p+1}\right)_{old} + \Delta t\left(S_h^{p+1}\right)_{old}\right)^{-1}\left(\left(M_h^{p+1}\right)_{old}\underline{h}^p + \Delta t\left(\underline{b}_h^{p+1}\right)_{old}\right),$$

$$\left(\underline{\rho}^{p+1}\right)_{new} = \left((M_\rho + \Delta t\left(S_\rho^{p+1}\right)_{old}\right)^{-1}\left(M_\rho\underline{\rho}^p + \Delta t\left(\underline{b}_\rho^{p+1}\right)_{old}\right).$$

In Figure 12, the relative difference $\frac{\underline{h}_1-\underline{h}_2}{\underline{h}_2}$ and $\frac{\underline{\rho}_1-\underline{\rho}_2}{\underline{\rho}_2}$ is shown, where $\underline{h}_1$ and $\underline{h}_2$ come respectively from semi-implicit and fully implicit solutions. A similar argument applies to $\underline{\rho}_1$ and $\underline{\rho}_2$. The difference is more noticeable where the gradient is large in actual variables.

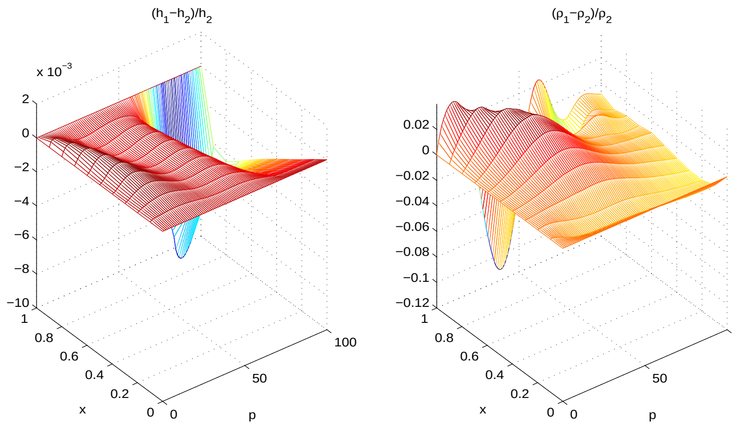

FIGURE 12. $\frac{\underline{h}_1-\underline{h}_2}{\underline{h}_2}$ and $\frac{\underline{\rho}_1-\underline{\rho}_2}{\underline{\rho}_2}$, $D = 1, q = 0, \Delta x = 0.1,\ h(x, 0) = 3 - x,\ h(0, t) = 2 + sin5t,\ h(1, t) = 2 + cost\rho(x, 0) = 1.5 - x,\ \rho(0, t) = 1.5$

5. Finite Elements in Two Dimensions

We have divided the two-dimensional unit domain into triangular elements. The mesh and the element topology is shown in Figure 13. Total number of elements in this case is $n = 2(n_x - 1)(n_y - 1)$, where n_x and n_y are the node count in respective directions. We have considered 2-node boundary elements. Total number of boundary elements are $2(n_x-1)+2(n_y-1)=2(n_x+n_y)-4$. For two-dimensional problems throughout this report, we maintained this convention for node and boundary numbering order.

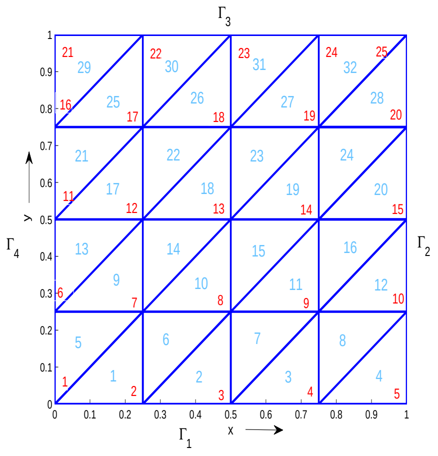

FIGURE 13. A 5x5 mesh, red numbers indicate nodes and blue number are the element labels. Γ_i are the non-overlapping boundary segments

Again, the linear basis functions $\phi(\underline{x})$ has been used. Where $\underline{x}=[x,\ y]^T$ and $\phi(\underline{x})$ is defined as,

$$\phi(\underline{x})=a+bx+cy, \tag{86}$$

$$\phi_i(\underline{x}_j)=\delta_{ij}. \tag{87}$$

For a typical element e_k, with nodes $\underline{x}_1$, $\underline{x}_2$, and $\underline{x}_3$, the basis function is determined by using equations (86) and (87), given by,

$$\begin{bmatrix} a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \\ c_1 & c_2 & c_3 \end{bmatrix}\begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_2 & y_2 \end{bmatrix}=\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{88}$$

where,

$$\underline{x}_k=[x_k,\ y_k]^T,\ k=1,2,3.$$

All the coefficients are determined by solving equation (88) and they are given by,

$$\begin{array}{rlrlrl} b_1 &= \frac{1}{\Delta}\left(y_2-y_3\right), & b_1 &= \frac{1}{\Delta}\left(y_3-y_1\right), & b_1 &= \frac{1}{\Delta}\left(y_1-y_2\right), \\ c_2 &= \frac{1}{\Delta}\left(x_3-x_2\right), & c_2 &= \frac{1}{\Delta}\left(x_1-x_3\right), & c_2 &= \frac{1}{\Delta}\left(x_2-x_1\right), \\ a_1 &= 1-b_1x_1-c_1y_1, & a_2 &= 1-b_2x_2-c_2y_2, & a_3 &= 1-b_3x_3-c_3y_3. \end{array} \tag{89}$$

Where $\Delta=(x_2-x_1)(y_3-y_2)-(y_2-y_1)(x_3-x_2)$, which also is twice the area of the triangle. Hence we require non-zero area for all elements. For the triangular elements, the following Newton-Cotes quadrature rule has been used:

$$\int_\Omega g(x)d\Omega=\frac{|\Delta|}{6}\left(g(\underline{x}_1)+g(\underline{x}_2)+g(\underline{x}_3)\right). \tag{90}$$

5.1. **Poisson Equation in Two Dimensions.** The steady-state diffusion equation in two dimensions is given by,

$$-\overrightarrow{\nabla}.\left(D\overrightarrow{\nabla}u\right)=f(\underline{x}),\quad \text{on } \Omega \tag{91}$$

$$\begin{array}{ll} u=2-x, & \text{at } \Gamma_1, \\ u=1+x, & \text{at } \Gamma_3, \\ \frac{\partial u}{\partial \hat{n}}=0, & \text{on } \Gamma_2\cup\Gamma_4, \end{array}$$

where,

$$\overrightarrow{\nabla}=\langle\frac{\partial}{\partial x},\frac{\partial}{\partial y}\rangle,$$

and $\hat{n}$ is the outward pointing unit normal vector. The computational domain is represented by the open region Ω, while $\Gamma=\cup_{i=1}^{4}\Gamma_i$ represents the domain boundary. The weak form of equation (91) is given by the following equation.

$$-\int_\Omega \overrightarrow{\nabla}.\left(D\overrightarrow{\nabla}u\right)\eta(\underline{x})d\Omega=\int_\Omega f(\underline{x})\eta(\underline{x})d\Omega. \tag{92}$$

Using Gauss Theorem, we have,

$$D\int_\Omega \overrightarrow{\nabla}u.\overrightarrow{\nabla}\eta(\underline{x})d\Omega-D\int_\Gamma\frac{\partial u}{\partial\hat{n}}\eta(\underline{x})d\Gamma=\int_\Omega f(\underline{x})\eta(\underline{x})d\Omega. \tag{93}$$

We take $\eta(\underline{x})=0$ for $\underline{x}\in\Gamma_1\cup\Gamma_3$. This makes the boundary integral $\int_\Gamma\frac{\partial u}{\partial\hat{n}}\eta(\underline{x})d\Gamma$ equal to zero because $\frac{\partial u}{\partial\hat{n}}=0$ for $\underline{x}\in\Gamma_2\cup\Gamma_4$. After eliminating the boundary integral, the discretized form of equation (93) is given by,

$$D\int_\Omega\sum_{j=1}^{n}u_j\overrightarrow{\nabla}\phi_j(\underline{x}).\overrightarrow{\nabla}\phi_i(\underline{x})d\Omega=\int_\Omega f(\underline{x})\phi_i d\Omega,$$
$$\forall i\in\{1,2,...,n\}\ \text{ for which } \underline{x}_i\notin\Gamma_1\cup\Gamma_3. \tag{94}$$

The element matrix S_e pertaining to the kth element e_k is computed as follows,

$$S_{e_{i,j}}=D\int_\Omega\overrightarrow{\nabla}\phi_j(\underline{x}).\overrightarrow{\nabla}\phi_i(\underline{x})d\Omega=\frac{|\Delta|}{6}\left(b_ib_j+c_ic_j\right),\ i,j=1,2,3. \tag{95}$$

Let $\underline{x}_{k_1},\underline{x}_{k_2}$, and $\underline{x}_{k_3}$ be three nodes of e_k. Then the stiffness matrix S is updated in the following way,

$$S_{k_i,k_j}=S_{k_i,k_j}+S_{e_{i,j}},\ i,j=1,2,3,\ k=1,2,...,n.$$

A similar procedure is followed for the right-hand side of equation (94) i.e., the element vector $\underline{f}_e$ is determined for all elements and the global vector $\underline{f}$ is updated accordingly. Equation (94) written in a matrix form is given by,

$$S_{n\times n}\underline{u}_n=\underline{f}_n. \tag{96}$$

Next we partition vectors and the matrix in the above equation such that all the known values in $\underline{u}_n$ could be taken to the right hand side of the equation. Let $n=m+r$ such that $\underline{u}_m$ are unknown values and $\underline{u}_r$ are given values (from essential boundary conditions) i.e., we rearrange equation (96) for rows and columns for this purpose. Equation (96) is rewritten as,

$$(S)_{m+r\times m+r}\,(\underline{u})_{m+r}\ =\ (\underline{f})_{m+r}, \tag{97}$$

or,

$$\begin{bmatrix} S_{m\times m} & S_{m\times r}\\ S_{r\times m} & S_{r\times r}\end{bmatrix}\begin{bmatrix}\underline{u}_m\\ \underline{u}_r\end{bmatrix}=\begin{bmatrix}\underline{f}_m\\ \underline{f}_r\end{bmatrix}. \tag{98}$$

From this equation we have,

$$S_{m\times m}\ \underline{u}_m+S_{m\times r}\ \underline{u}_r=\underline{f}_m,$$

or,

$$S_{m\times m}\ \underline{u}_m = \underline{f}_m - S_{m\times r}\ \underline{u}_r,$$

or,

$$\underline{u} = S^{-1}\left(\underline{f} + \underline{b}\right),$$

where $\underline{b} = -S_{m\times r}\ \underline{u}_r$

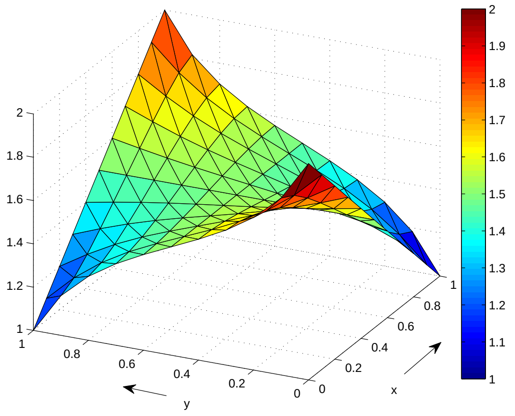

FIGURE 14. The numerical solution of the diffusion equation, $D = 1, f = 1$

5.2. **Convection-Diffusion Equation in Two-Dimension.** Given the problem,

$$-\vec{\nabla}.\left(D\vec{\nabla}u\right) + \vec{v}.\vec{\nabla}u = 0, \tag{99}$$

$$\begin{aligned} u &= 1, && \text{at } \Gamma_1,\\ u &= 2, && \text{at } \Gamma_3,\\ \tfrac{\partial u}{\partial \hat{n}} &= 0, && \text{at } \Gamma_2\cup\Gamma_4. \end{aligned}$$

Let S_e^D and S_e^C be element matrices due to diffusion and convection terms respectively. We have computed S_e^D in the previous section and the other part is given by,

$$S^C_{e_{i,j}} = \vec{v}.\int_\Omega \vec{\nabla}\phi_j(\underline{x})\phi_i(\underline{x})d\Omega = \frac{|\Delta|}{6}\left(v_x b_j + v_y c_j\right),\ i,j = 1,2,3. \tag{100}$$

The numerical solution of equation (99) is shown in Figure 15. Now boundary

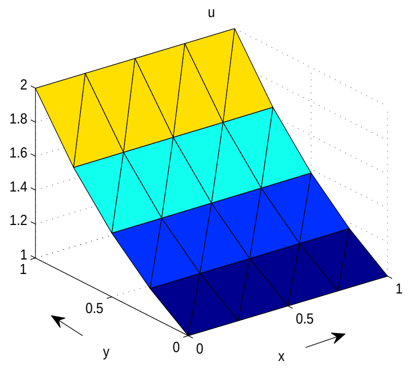

FIGURE 15. Numerical Solution of Steady-State Convection-Diffusion equation, $D = 1$, $n_x = n_y = 5$

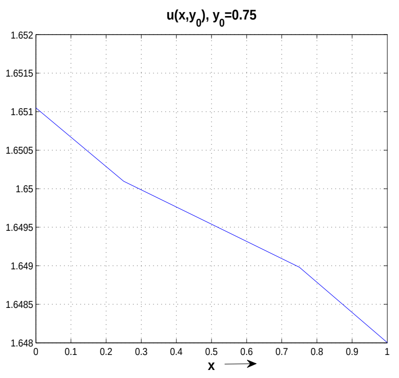

FIGURE 16. Graph of $u(x, y_0)$, $y_0 = 0.75$) expected to be constant, $D = 1$, $n_x = n_y = 5$

conditions are such that all variations are in the y-direction and we expect a constant solution in the x-direction. In Figure 16 we have shown the solution $u(x, y_0)$, $y_0 = 0.75$. We observe a small variation even in the x-direction. In order to investigate the cause of this, we divide our domain into two elements, shown in Figure 17 and take the following boundary conditions,

$$\begin{aligned} u &= 1, && \text{at } \Gamma_1, \\ \frac{\partial u}{\partial \hat{n}} &= 1, && \text{at } \Gamma_3, \\ \frac{\partial u}{\partial \hat{n}} &= 0, && \text{at } \Gamma_2 \cup \Gamma_4. \end{aligned}$$

Only two element matrices S_{e_1} and S_{e_2} are to be determined in this case. Taking

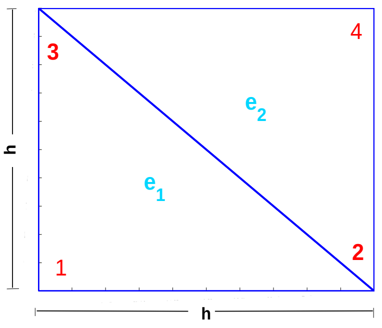

FIGURE 17. Computational Domain with two elements

$v_x = 0$ everywhere, we have the element matrix due to the convection part,

$$S^C_{e_{i,j}} = \frac{|\Delta|}{6} v_y c_j, \;\; i, j = 1, 2, 3, \;\; \text{(first element)}. \tag{101}$$

It is clear from equation (89) that c_j can be expressed in terms of h and Δ. Hence we can write,

$$S^C_{e_{i,j}} = h\beta_j, \tag{102}$$

where h is defined in Figure 17 and $\beta_j = v_y \frac{\Delta c_j}{6h}$ is independent of the element dimension h and the area $\frac{\Delta}{2}$ (because $c_j \propto \frac{h}{\Delta}$). The complete element matrix for the first element e_1 is given by,

$$S^C_{e_1} = \begin{bmatrix} h\beta_1 & h\beta_2 & h\beta_3 \\ h\beta_1 & h\beta_2 & h\beta_3 \\ h\beta_1 & h\beta_2 & h\beta_3 \end{bmatrix} = \frac{v_y}{6} \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}. \tag{103}$$

With current topology β_2 is always zero (regardless of h and Δ values). We update the global matrix S^C and it is given by,

$$S^C = \begin{bmatrix} h\beta_1 & 0 & 0 & h\beta_3 \\ h\beta_1 & 0 & 0 & h\beta_3 \\ 0 & 0 & 0 & 0 \\ h\beta_1 & 0 & 0 & h\beta_3 \end{bmatrix}. \tag{104}$$

Similarly $S^C_{e_2}$ and the updated S^C are given by,

$$S^C_{e_2} = \begin{bmatrix} h\beta_1 & h\beta_3 & 0 \\ h\beta_1 & h\beta_3 & 0 \\ h\beta_1 & h\beta_3 & 0 \end{bmatrix} = \frac{v_y}{6} \begin{bmatrix} -1 & 1 & 0 \\ -1 & 1 & 0 \\ -1 & 1 & 0 \end{bmatrix}, \tag{105}$$

$$S^C = \begin{bmatrix} 2h\beta_1 & 0 & h\beta_3 & h\beta_3 \\ h\beta_1 & 0 & 0 & h\beta_3 \\ h\beta_1 & 0 & h\beta_3 & 0 \\ 2h\beta_1 & 0 & h\beta_3 & h\beta_3 \end{bmatrix}. \tag{106}$$

Now we come to the diffusion part;

$$S^D_{e_1} = \frac{\Delta}{2}\left(b_i b_j + c_i c_j\right), \ i, j = 1, 2, 3. \tag{107}$$

From equations (89) and (107), it is obvious that $S^D_{e_1}$ is independent of h and Δ. The diffusion part of the stiffness matrix (computed from $S^D_{e_1}$ and $S^D_{e_2}$) is given by,

$$S^D = \frac{1}{2} \begin{bmatrix} 2 & -1 & -1 & 0 \\ -1 & 2 & 0 & -1 \\ -1 & 0 & 2 & -1 \\ 0 & -1 & -1 & 2 \end{bmatrix}, \tag{108}$$

which is symmetric about both diagonals. Therefore we can write it in the following form;

$$S^D = \begin{bmatrix} d_1 & d_2 & d_3 & d_4 \\ d_2 & d_1 & d_4 & d_3 \\ d_3 & d_4 & d_1 & d_2 \\ d_4 & d_3 & d_2 & d_1 \end{bmatrix}, \tag{109}$$

where $d_1 = 1$, $d_2 = -\frac{1}{2}$, $d_3 = -\frac{1}{2}$, and $d_4 = 0$. The stiffness matrix is given by the following equation.

$$S = S^D + S^C = \begin{bmatrix} 2h\beta_1 + d_1 & d_2 & h\beta_3 + d_3 & h\beta_3 + d_4 \\ h\beta_1 + d_2 & d_1 & d_4 & h\beta_3 + d_3 \\ h\beta_1 + d_3 & d_4 & h\beta_3 + d_1 & d_2 \\ 2h\beta_1 + d_4 & d_3 & h\beta_3 + d_2 & h\beta_3 + d_1 \end{bmatrix}. \tag{110}$$

The vector $\underline{b}$ is determined as,

$$\begin{bmatrix} b_3 \\ b_4 \end{bmatrix} = \begin{bmatrix} S_{3,1} & S_{3,2} \\ S_{4,1} & S_{4,2} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}, \tag{111}$$

or,

$$\begin{bmatrix} b_3 \\ b_4 \end{bmatrix} = \begin{bmatrix} h\beta_1 + d_3 & d_4 \\ 2h\beta_1 + d_4 & d_3 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} h\beta_1 + d_3 + d_4 \\ 2h\beta_1 + d_3 + d_4 \end{bmatrix}. \tag{112}$$

Now unknown values u_3 and u_4 are determined as,

$$\begin{bmatrix} u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} S_{3,3} & S_{3,4} \\ S_{4,3} & S_{4,4} \end{bmatrix}^{-1} \left(\begin{bmatrix} b_3 \\ b_4 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} \right), \tag{113}$$

where the last vector on the right hand side comes from natural boundary conditions,

$$\begin{bmatrix} u_3 \\ u_4 \end{bmatrix} = \frac{1}{\Delta_s} \begin{bmatrix} h\beta_3 + d_1 & -d_2 \\ -h\beta_3 - d_2 & h\beta_3 + d_1 \end{bmatrix} \begin{bmatrix} h\beta_1 + d_3 + d_4 + 1 \\ 2h\beta_1 + d_3 + d_4 + 1 \end{bmatrix}, \tag{114}$$

where $\Delta_s = (h\beta_3 + d_1)(h\beta_3 + d_1) - d_2(h\beta_3 + d_2)$.

$$\begin{bmatrix} u_3 \\ u_4 \end{bmatrix} = \frac{1}{\Delta_s} \begin{bmatrix} (h\beta_3 + d_1)(h\beta_1 + d_3 + d_4 + 1) - d_2(2h\beta_1 + d_3 + d_4 + 1) \\ (-h\beta_3 - d_2)(h\beta_1 + d_3 + d_4 + 1) + (h\beta_3 + d_1)(2h\beta_1 + d_3 + d_4 + 1) \end{bmatrix}. \tag{115}$$

Finally we check for the variation the in x-direction;

$$\Delta u = u_4 - u_3 = \frac{h}{\Delta_s}\left[h\beta_1\beta_3 + \beta_1 d_2 - \beta_3(h\beta_1 + d_3 + d_4 + 1)\right], \tag{116}$$

as $h \to 0$, $\Delta_s \to d_1^2 - d_2^2$ and $\Delta u \to 0$. It is clear from equation (116) that we can not have constant numerical solution in x-direction with given conditions. This effect can be reduced by using a finer mesh.

Further more, we consider only the diffusion part and set $v_y = 0$. That makes $\beta_i = 0$ ($\beta_i = \frac{\Delta}{6h} v_y c_i$). From equation (116) we have,

$$u_4 - u_3 = 0, \tag{117}$$

i.e., we have no error in the x-direction.

6. FEM Model of Darcy Flow in Porous Medium

The system of non-linear equations is reconsidered here in two dimensions.

$$\frac{\partial(\rho h)}{\partial t} + \overrightarrow{\nabla}.(\rho h \overrightarrow{v}) = \overrightarrow{\nabla}.\left(D_1 \overrightarrow{\nabla} T\right) + q(\underline{x}, t), \tag{118}$$

$$\frac{\partial \rho}{\partial t} + \overrightarrow{\nabla}(\rho \overrightarrow{v}) = 0, \tag{119}$$

$$\overrightarrow{v}(\underline{x}, t) = -\lambda_1 \overrightarrow{\nabla} P \tag{120}$$

$$P(\underline{x}, t).V = RT(\underline{x}, t), \tag{121}$$

$$h(\underline{x}, t) = cT(\underline{x}, t). \tag{122}$$

Initial and boundary conditions are also known. We apply the same procedure as given in the section for system of equations in one dimension. The above system can be reduced to the following two equations in two variables (without a source term),

$$\rho \frac{\partial h}{\partial t} - \lambda \rho \overrightarrow{\nabla} h.\overrightarrow{\nabla} h = \overrightarrow{\nabla}.\left(D \overrightarrow{\nabla} h\right), \tag{123}$$

$$\frac{\partial \rho}{\partial t} - \lambda \rho \overrightarrow{\nabla}.\overrightarrow{\nabla} h - \lambda \overrightarrow{\nabla} h.\overrightarrow{\nabla} \rho = 0. \tag{124}$$

We take the following initial and boundary conditions,

$$h(\underline{x}, t = 0) = 1 - \frac{x}{2} - \frac{y}{2}$$

$$\rho(\underline{x}, t = 0) = 1 - \frac{x}{2} - \frac{y}{2}$$

$$h = \begin{cases} 1 - \frac{x}{2} & \text{at } \Gamma_1, \\ \frac{1}{2} - \frac{y}{2}, & \text{at } \Gamma_2, \\ \frac{1}{2} - \frac{x}{2}, & \text{at } \Gamma_3, \\ 1 - \frac{y}{2} & \text{at } \Gamma_4. \end{cases}$$

$$\rho = \begin{cases} 1 - \frac{x}{2} & \text{at } \Gamma_1, \\ 1 - \frac{y}{2} & \text{at } \Gamma_4. \end{cases}$$

Where Γ_1, and Γ_1 are inflow boundaries. The time discretization of equation (123) and (124) is done as follows (semi-implicit scheme),

$$\rho^p \frac{h^{p+1} - h^p}{\Delta t} - \lambda \rho^p \overrightarrow{\nabla} h^p . \overrightarrow{\nabla} h^{p+1} = \overrightarrow{\nabla} . \left(D \overrightarrow{\nabla} h^{p+1} \right), \tag{125}$$

$$\frac{\rho^{p+1} - \rho^p}{\Delta t} - \lambda \rho^{p+1} \overrightarrow{\nabla} . \overrightarrow{\nabla} h^p - \lambda \overrightarrow{\nabla} h^p . \overrightarrow{\nabla} \rho^{p+1} = 0. \tag{126}$$

We have used Standard Galerkin Approximation for both equations. The element stiffness matrix for equation (125) is computed as follows (the diffusion part $S^D_{h_e}$ is calculated in a previous section),

$$S_{h_{e_{i,j}}} = -\lambda \int_{e_m} \sum_{k=1}^{3} \rho^p_k \phi_k \sum_{l=1}^{3} h^p_l \overrightarrow{\nabla} \phi_l . \left(\overrightarrow{\nabla} \phi_j \right) \phi_i d\Omega \ + \ S^D_{h_{e_{i,j}}}, \qquad m = 1, 2..., n,$$

$$S_{h_{e_{i,j}}} = -\lambda \int_{e_m} \sum_{k=1}^{3} \rho^p_k \phi_k \sum_{l=1}^{3} h^p_l \left(b_j b_l + c_j c_l \right) \phi_i d\Omega \ + \ S^D_{h_{e_{i,j}}}.$$

$$S_{h_{e_{i,j}}} = -\lambda \frac{|\Delta|}{6} \rho^p_i \sum_{l=1}^{3} h^p_l \left(b_j b_l + c_j c_l \right) \ + \ S^D_{h_{e_{i,j}}}.$$

The element mass matrix is given by,

$$M_{h_{e_{i,j}}} = \int_{e_m} \sum_{k=1}^{3} \rho^p_k \phi_k \phi_j \phi_i d\Omega = \frac{|\Delta|}{6} \rho^p_i \delta_{ij}. \tag{127}$$

Similarly element matrices for equation (126) are given by,

$$S_{\rho_{e_{i,j}}} = -\lambda \frac{|\Delta|}{6} \sum_{l=1}^{3} h^p_l \left(b_j b_l + c_j c_l \right), \tag{128}$$

$$M_{\rho_{e_{i,j}}} = \frac{|\Delta|}{6} \delta_{ij}. \tag{129}$$

The computation of $\underline{b}$ is explained in the section for the diffusion problem and the matrix equations have the same notation as given for one-dimensional case, i.e.,

$$\underline{h}^{p+1} = \left(M^p_h + \Delta t S^p_h \right)^{-1} \left(M_h \underline{h}^p + \Delta t \underline{b}^p_h \right),$$
$$\underline{\rho}^{p+1} = \left(M_\rho + \Delta t S^p_\rho \right)^{-1} \left(M_\rho \underline{\rho}^p + \Delta t \underline{b}^p_\rho \right).$$

The numerical solution of h and ρ is given in Figure 18. In Figure 19 the function ρ is given at different iterations. We observe that it reaches a steady-state. This was expected because the boundary values are independent of t. The matrix equations for the fully implicit scheme are given by,

$$\left(\underline{h}^{p+1} \right)_{new} = \left(\left(M^{p+1}_h \right)_{old} + \Delta t \left(S^{p+1}_h \right)_{old} \right)^{-1} \left(\left(M^{p+1}_h \right)_{old} \underline{h}^p + \Delta t \left(\underline{b}^{p+1}_h \right)_{old} \right),$$

$$\left(\underline{\rho}^{p+1} \right)_{new} = \left(\left(M_\rho + \Delta t \left(S^{p+1}_\rho \right)_{old} \right)^{-1} \left(M_\rho \underline{\rho}^p + \Delta t \left(\underline{b}^{p+1}_\rho \right)_{old} \right).$$

The numerical result in this case is given in Figure 20.

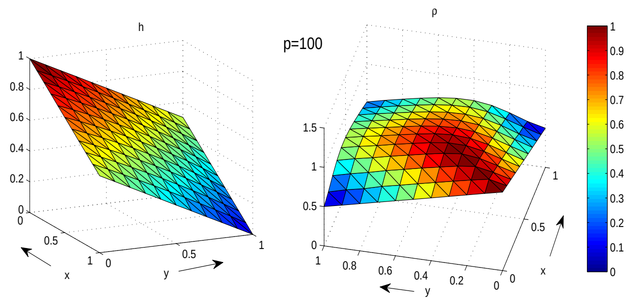

FIGURE 18. The Numerical Solution of h and ρ with Semi-Implicit Scheme, $D = 1$, $\Delta t = 0.01$, $p = 100$.

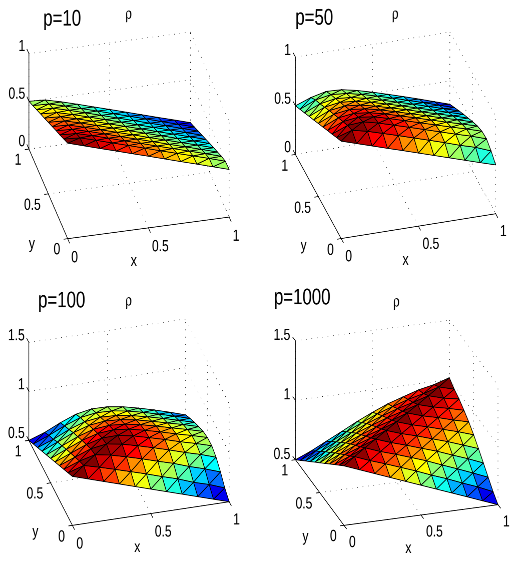

FIGURE 19. The Numerical Solution of ρ (with Semi-Implicit Scheme) at different iterations p.

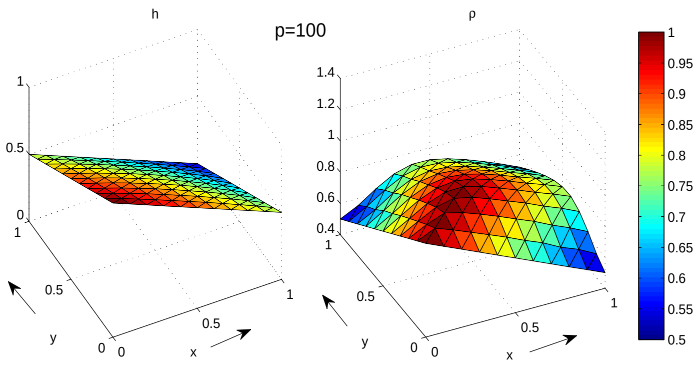

FIGURE 20. The Numerical Solution of h and ρ with Fully Implicit Scheme, $D = 1$, $\Delta t = 0.01$, $p = 100$.

7. CONCLUSIONS

In this report we computed numerical solutions starting with steady-state linear problem in one dimension and ending with a system of coupled non-linear equations in two dimensions. We observed a good agreement in numerical and analytical results where the comparison was possible to make. Initial and boundary conditions effect the accuracy and stability of the solution, especially when they are discontinuous or the gradient is large. For non-linear problems, the Picard method worked reasonably well. For further work, few of the equations in the system we treated, should be replaced with more realistic modeling. The non-trivial problem domains should also be considered. The accuracy and the stability issues are also there for the said system. Furthermore, the said system should be discretized in conservative form so that numerical results should reflect the physical laws, realistically.

References

[1] Klaus A. Hoffmann, "Computational Fluid Dynamics, Vol I", 4e, EES, Wichita, USA, 2000

[2] J. van Kan, A Segal, F. Vermolen, "Numerical Methods in Scientific Computing", VSSD, 2004

[3] Randall J. Leveque, "Finite Volume Methods for Hyperbolic Problems", Cambridge University Press, USA, 2002.

[4] J.N. Reddy, "An Introduction to the Finite Element Method", 2e, McGraw-Hill, 1993.